UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/716,093	11/18/2003	Michael C. Tulkoff	VIGN1660-1	4856

44654 7590 04/14/2009

SPRINKLE IP LAW GROUP
1301 W. 25TH STREET
SUITE 408
AUSTIN, TX 78705

EXAMINER	
SAEED, USMAAN	
ART UNIT	PAPER NUMBER
2166	

MAIL DATE	DELIVERY MODE
04/14/2009	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/716,093	TULKOFF ET AL.	
	Examiner	Art Unit	
	USMAAN SAEED	2166	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *26 January 2009*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *49,51-60 and 65-73* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *49,51-60 and 65-73* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *18 November 2003* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

Continued Examination Under 37 CFR 1.114

1. A request for continued examination under 37 CFR 1.114, including the fee set forth in 37 CFR 1.17(e), was filed in this application after final rejection. Since this application is eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e) has been timely paid, the finality of the previous Office action has been withdrawn pursuant to 37 CFR 1.114. Applicant's submission filed on 01/26/2009 has been entered.

Claim Rejections - 35 USC § 101

2. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

Claims 49, 51-60, and 65-69 is rejected under 35 U.S.C. 101 as being directed to non-statutory subject matter. The language of the claim raises a question as to whether the claim is directed merely to an environment or machine which would result in a practical application producing a concrete useful, and tangible result to form the basis of statutory subject matter under 35 U.S.C. 101.

Claims 49, 51-60, and 65-69 are rejected because the method claims do not qualify as a statutory process. These claims are not statutory because a process must be tied to another statutory class. Thus to qualify as a statutory process, the claims should positively recite the other statutory class to which it is tied, for example by identifying the apparatus that accomplished the method steps.

To expedite a complete examination of the instant application the claims rejected under U.S.C. 101 (nonstatutory) above are further rejected as set forth below in anticipation of application amending these claims to place them within the four categories of invention.

Claim Rejections - 35 USC § 103

3. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

This application currently names joint inventors. In considering patentability of the claims under 35 U.S.C. 103(a), the examiner presumes that the subject matter of the various claims was commonly owned at the time any inventions covered therein were made absent any evidence to the contrary. Applicant is advised of the obligation under 37 CFR 1.56 to point out the inventor and invention dates of each claim that was not commonly owned at the time a later invention was made in order for the examiner to consider the applicability of 35 U.S.C. 103(c) and potential 35 U.S.C. 102(e), (f) or (g) prior art under 35 U.S.C. 103(a).

Claims 49, 51-52, 56-59 65, 68-70, and 73 are rejected under 35 U.S.C. 103(a) as being unpatentable over **Macleod et al.** (**Macleod** hereinafter) (US PG Pub No. 2003/0105770) in view of **Dennis A. VanDusen** (**VanDusen** hereinafter) (U.S. PG Pub No.

2003/0208397), further in view of **Savage et al. (Savage** hereinafter) (U.S. Patent No. 6,604,110).

With respect to claim 49, **Macleod** teaches **a method for integrating data into a content management system, comprising:**

"analyzing a set of data and generating a set of content types to represent the set of data in the content management system based on the analysis of the data" as a content class models a set of items that have similar properties and fulfill similar purposes. A content class defines the purpose or content of an item by containing as its elements a list of properties appropriate for that purpose or content **(Macleod** Paragraph 0022). The content class includes a flexible attribute having a data type. The directory schema, multiple instances of the same object class can have attributes that provide completely different data types and completely different data operations **(Macleod** Abstract). All the classes are being analyzed and each content class has items with similar properties and has a data type.

"saving the set of content types in a memory" Schema definition require that objects conform to fixed data formats of classes defined in the directory schema. In other words, for example, if a class consists of ten (10) data elements, then any object that is based on that class will require the data storage to store those 10 data elements, regardless of whether each of the 10 elements even contain any data **(Macleod** Paragraph 0055).

"generating a set of content type objects corresponding to the set of content types, wherein a content type object is an instantiation of a content type embodied in the content management system" as a content class models a set of items that have similar

properties and fulfill similar purposes. A content class defines the purpose or content of an item by containing as its elements a list of properties appropriate for that purpose or content (**Macleod** Paragraph 0022). Schema definition require that objects conform to fixed data formats of classes defined in the directory schema. In other words, for example, if a class consists of ten (10) data elements, then any object that is based on that class will require the data storage to store those 10 data elements, regardless of whether each of the 10 elements even contain any data (**Macleod** Paragraph 0055).

Further Macloed1 teaches the act of creating an object of a particular class (or "data type") is called "instantiation" of the particular class, thereby creating an "object instance" of the class (**Macleod** Paragraph 0025). Examiner interprets object of a particular class or data type as a content type object.

"generating a set of content instance objects from the content type objects wherein each content instance object is an instantiation of a content instance and is associated with a content type object or a content type" as FIG. 6 shows an exemplary procedure 600 to change the operational or data providing nature of multiple object instances of a base content class in a directory schema independent of modifying the directory schema. At block 610, the procedure instantiates a first object instance of a flexible content class 422 (**Macleod** Paragraph 0074).

Further Macloed teaches the act of creating an object of a particular class (or "data type") is called "instantiation" of the particular class, thereby creating an "object instance" of the class (**Macleod** Paragraph 0025).

"associating each of the set of data with at least one of the content instance object, wherein at least one content instance object is associated with two or more datum of the set of data" as the procedure assigns a first data string (e.g., XML) to a flexible attribute 418 in the first flexible object instance (block 610), the first data string defines any combination of a first operational and a data providing nature of the first object instance. Specifically, an application that has instantiated or that is using the first object instance knows of the first object instance's interface and how to unpack and use the first data string (**Macleod** Paragraph 0075). Multiple instances of the same object class can have attributes that provide completely different data types and completely different data operations (**Macleod** Abstract).

Further, **Macleod** teaches for instance, consider that an application can assign the flexible attribute in the first instantiated object to have any combination of one or more data types (e.g., integer, real, string, floating, character, and so on), or operational properties (e.g., an operation can be defined to do just about anything imaginable such as to send an e-mail message, to report statistics, to manage a rocket launch, and so on). Whereas the flexible attribute in the second instance of the object can be assigned completely different properties that are independent of any characteristics of the data types or operations that correspond to the flexible attribute of the first instance of the object (**Macleod** Paragraph 0077 and figure 6).

"each of the datum residing in a distinct storage" as (**Macleod** Figure 5). FIG. 5 shows an exemplary system 500 to implement a directory schema 400 of FIG. 4 with flexible structural content classes and attributes. Figure 5 also shows databases or directories and the only objects that can be represented in the distributed directory are those that meet the content class qualifications specified by the directory schema.

"managing the set of data residing in the repository using the content instance objects of the content management system, wherein the two or more datum are managed as a single entity using the at least one content instance object" as a system to generate and manage objects based on an exemplary directory schema 400 with flexible attributes 418 may be implemented **(Macleod** Paragraph 0087). FIG. 6 shows an exemplary procedure 600 to change the operational or data providing nature of multiple object instances of a base content class in a directory schema independent of modifying the directory schema.

Macleod teaches the elements of claim 49 as noted above but does not explicitly discloses **"analyzing a set of data and generating a set of content types," "wherein one of the content types comprise a policy annotation, the policy annotation comprising management information including workflow corresponding to the content type," "generating a set of content type objects corresponding to the set of content types" and "legacy data residing in a legacy repository external to said content management system."**

However, **VanDusen** discloses **"analyzing a set of data and generating a set of content types"** as a key component of a broad content management solution is the strategic use of metadata, or what we call site categories. Metadata is used to tag, or mark content items with data that identifies the types of content in use **(VanDusen** Paragraphs 0078, 0061). Using content delivery configuration tools, site affiliates can define what types of content and what rules to use to provide that content **(VanDusen** Paragraph 0171). Metadata is being used in the analysis of data and it identifies the types of data in use.

"wherein one of the content types comprise a policy annotation, the policy annotation comprising management information including workflow corresponding to the content type" as a business manager can simply apply a new policy to a content type, and the system will automatically enforce this business rule. For example, a business manager might define a set of policies for controlling the management and creation of product promotions (**VanDusen** Paragraph 0133 and 0276).

"generating a set of content type objects corresponding to the set of content types" as a site object model is a collection of ContentObject Types. Each object type contains a set of properties and methods. Properties are the elements of data associated with an object type (**VanDusen** Paragraph 0061).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teaching of the cited references because **VanDusen's** teachings would have allowed **Macleod** to provide metadata which provides the foundation for a range of other capabilities, including higher level end user browsing and searching tools, user interest profiling, and rich reporting on which classes of content and products are most popular in a Web site. Metadata is also used to tag, or mark content items with data that identifies the types of content in use.

Macleod and VanDusen teach the elements of claim 49 as noted above but do not explicitly teach **"legacy data residing in a legacy repository external to said content management system."**

However, **Savage** teaches **"analyzing a set of legacy data residing in a legacy repository external to said content management system"** as analyze the source databases to identify the type and form of the source data and the structure and operational system of the source databases, to determine their relevance to the EDM target database; (ii) load the source data from the legacy databases to an intermediate database using extract, transform, and load (ETL) software tools or hand-coded programs; (iii) massage, cleanse, and manipulate ("transform") the source data in the intermediate database into the form needed for the EDM application; and (iv) load the transformed source data into the EDM application, and format it in those schemas necessary for OLAP or other EDM applications (**Savage** Col 2, Lines 8-19).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teaching of the cited references because **Savage's** teachings would have allowed **Macleod and VanDusen** to provide a generic storage model that is suitable as a data source from which enterprise data management applications may be generated in any desired model, format, or language.

With respect to claim 51, **Macleod** teaches, **"generating the content type comprises specifying attributes associated with the content type"** as a directory schema with object classes that have flexible attributes. The content class includes a flexible attribute having a data type (**Macleod** Paragraph 0012 & 0054).

With respect to claim 52, **Macleod** teaches **"for each of the set of the content types, analyzing the data to obtain a first set of the data corresponding to the content type"** as schema definition require that objects conform to fixed data formats of classes defined in the directory schema. In other words, for example, if a class consists of ten (10) data elements, then any object that is based on that class will require the data storage to store those 10 data elements, regardless of whether each of the 10 elements even contain any data (**Macleod** Paragraph 0055). The first instantiated object to have any combination of one or more data types (e.g., integer, real, string, floating, character, and so on), or operational properties (e.g., an operation can be defined to do just about anything imaginable such as to send an e-mail message, to report statistics, to manage a rocket launch, and so on). Whereas the flexible attribute in the second instance of the object can be assigned completely different properties that are independent of any characteristics of the data types or operations that correspond to the flexible attribute of the first instance of the object (**Macleod** Paragraph 0077).

Macleod teaches the elements of claim 52 as noted above but does not explicitly discloses **"legacy data."**

However, **Savage** teaches **"legacy data"** (**Savage** Col 2, Lines 8-19).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teaching of the cited references because **Savage's** teachings would have allowed **Macleod and VanDusen** to provide a generic storage model that is suitable as a data source from which enterprise data management applications may be generated in any desired model, format, or language.

With respect to claim 56, **Macleod** teaches **"wherein each of the set of content type objects is a structured definition of the corresponding content type"** as all directory schema 400 structural objects (other than "top") inherit properties from the class schema class 412. Structural content classes (with the exception of the "top" content class) include only those attributes that are defined by the attribute schema class 414 or those attributes defined by content classes that have been derived from the attribute schema class 414 (**Macleod** Paragraph 0029).

With respect to claim 57, **Macleod** teaches **"wherein each of the content type objects is an XML document"** as an application using an object instance that includes the flexible attribute can store, for example, an XML string on the flexible attribute property "attributeSyntax" (**Macleod1** Paragraph 0054). For example, consider the first XML string or document "<a> Data " (**Macleod1** Paragraph 0084).

With respect to claim 58 and 59, **Macleod** teaches, **"wherein each of the set of content types have associated workflows, access controls or policies and managing the set of data using workflows, access controls or policies associated with each of set of content types"** as a content class models a set of items that have similar properties and fulfill similar purposes. A content class defines the purpose or content of an item by containing as its elements a list of properties appropriate for that purpose or content (**Macleod** Paragraph 0022). A system to generate and manage objects based on an exemplary directory schema 400 with flexible attributes 418 may be implemented (**Macleod** Paragraph 0087).

Macleod teaches the elements of claim 58 and 59 as noted above but does not explicitly discloses **"workflows, access controls or policies" and "legacy data."**

However, **VanDusen** discloses **"workflows, access controls or policies"** as a business manager can simply apply a new policy to a content type, and the system will automatically enforce this business rule. For example, a business manager might define a set of policies for controlling the management and creation of product promotions (**VanDusen** Paragraph 0133 and 0276).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teaching of the cited references because **VanDusen's** teachings would have allowed **Macleod** to provide metadata which provides the foundation for a range of other capabilities, including higher level end user browsing and searching tools, user interest profiling, and rich reporting on which classes of content and products are most popular in a Web site. Metadata is also used to tag, or mark content items with data that identifies the types of content in use.

Macleod and VanDusen teach the elements of claim 59 as noted above but do not explicitly discloses **"legacy data."**

However, **Savage** teaches **"legacy data"** as (**Savage** Col 2, Lines 8-19).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teaching of the cited references because **Savage's** teachings would have allowed **Macleod and VanDusen** to provide a generic storage model that is suitable as a data source from which enterprise data management applications may be generated in any desired model, format, or language.

With respect to claim 65 **Macleod** teaches **a method for integrating legacy data into a content management system, comprising:**

"locating data to be integrated into said content management system" as the directory service 516 stores information about objects such as computers and resources on a network and makes this information available to users and network administrators in a directory that ties disparate databases 506, or "directories" of data together into a single, logical directory, or "metadirectory". Specifically, the directory service manages and maintains a distributed directory that is based on the directory schema 400 with flexible attributes (**Macleod** Paragraph 0072).

"creating a content instance object for each piece of data which matches a content type" " as FIG. 6 shows an exemplary procedure 600 to change the operational or data providing nature of multiple object instances of a base content class in a directory schema independent of modifying the directory schema. At block 610, the procedure instantiates a first object instance of a flexible content class 422 (**Macleod** Paragraph 0074). Further Macloed1 teaches the act of creating an object of a particular class (or "data type") is called "instantiation" of the particular class, thereby creating an "object instance" of the class (**Macleod** Paragraph 0025).

"associating or attaching said content instance object to said piece of data which said content instance object represents" as the procedure assigns a first data string (e.g., XML) to a flexible attribute 418 in the first flexible object instance (block 610), the first data string defines any combination of a first operational and a data providing nature of the first

object instance. Specifically, an application that has instantiated or that is using the first object instance knows of the first object instance's interface and how to unpack and use the first data string (**Macleod** Paragraph 0075). Multiple instances of the same object class can have attributes that provide completely different data types and completely different data operations (**Macleod** Abstract).

Further, **Macleod** teaches for instance, consider that an application can assign the flexible attribute in the first instantiated object to have any combination of one or more data types (e.g., integer, real, string, floating, character, and so on), or operational properties (e.g., an operation can be defined to do just about anything imaginable such as to send an e-mail message, to report statistics, to manage a rocket launch, and so on). Whereas the flexible attribute in the second instance of the object can be assigned completely different properties that are independent of any characteristics of the data types or operations that correspond to the flexible attribute of the first instance of the object (**Macleod** Paragraph 0077 and figure 6).

"persisting said content instance object in said content management system" as (**Macleod** figure 5).

"managing said piece of data using said content instance object" as a system to generate and manage objects based on an exemplary directory schema 400 with flexible attributes 418 may be implemented (**Macleod** Paragraph 0087)." FIG. 6 shows an exemplary procedure 600 to change the operational or data providing nature of multiple object instances of a base content class in a directory schema independent of modifying the directory schema.

Macleod teaches the elements of claim 65 as noted above but does not explicitly discloses **"enabling a user to define content types in terms of said user's business context" "creating a content type object for each of said content types" and "legacy data residing in a legacy repository external to said content management system."**

However, **VanDusen** teaches **"enabling a user to define content types in terms of said user's business context"** as when business users add new content items to the system, they apply site category metadata information to the items (**VanDusen** Paragraph 0080). A business manager can simply apply a new policy to a content type and the system will automatically enforce this business rule. For example, a business manager might define a set of policies for controlling the management and creation of product promotions (**VanDusen** Paragraph 0133).

"creating a content type object for each of said content types" as a site object model is a collection of ContentObject Types. Each object type contains a set of properties and methods. Properties are the elements of data associated with an object type (**VanDusen** Paragraph 0061).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teaching of the cited references because **VanDusen's** teachings would have allowed **Macleod** to provide metadata which provides the foundation for a range of other capabilities, including higher level end user browsing and searching tools, user interest profiling, and rich reporting on which classes

of content and products are most popular in a Web site. Metadata is also used to tag, or mark content items with data that identifies the types of content in use.

Macleod and VanDusen teach the elements of claim 65 as noted above but do not explicitly teach **"legacy data residing in a legacy repository external to said content management system."**

However, **Savage** teaches **"legacy data residing in a legacy repository external to said content management system"** as analyze the source databases to identify the type and form of the source data and the structure and operational system of the source databases, to determine their relevance to the EDM target database; (ii) load the source data from the legacy databases to an intermediate database using extract, transform, and load (ETL) software tools or hand-coded programs; (iii) massage, cleanse, and manipulate ("transform") the source data in the intermediate database into the form needed for the EDM application; and (iv) load the transformed source data into the EDM application, and format it in those schemas necessary for OLAP or other EDM applications (**Savage** Col 2, Lines 8-19).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teaching of the cited references because **Savage's** teachings would have allowed **Macleod and VanDusen** to provide a generic storage model that is suitable as a data source from which enterprise data management applications may be generated in any desired model, format, or language.

Claim 70 is essentially the same as claim 65 except that it sets forth the claimed invention as a medium carrying instruction and is rejected for the same reasons as applied hereinabove.

With respect to claim 69, **Macleod** does not explicitly teaches **"enabling said user to annotate policy information in defining said content types."**

However, **VanDusen** discloses **enabling said user to annotate policy information in defining said content types"** as a business manager can simply apply a new policy to a content type, and the system will automatically enforce this business rule. For example, a business manager might define a set of policies for controlling the management and creation of product promotions **(VanDusen** Paragraph 0133 and 0276).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teaching of the cited references because **VanDusen's** teachings would have allowed **Macleod** to provide metadata which provides the foundation for a range of other capabilities, including higher level end user browsing and searching tools, user interest profiling, and rich reporting on which classes of content and products are most popular in a Web site. Metadata is also used to tag, or mark content items with data that identifies the types of content in use.

With respect to claims 68 and 73, **Macleod and VanDusen** do not explicitly teach **"wherein said legacy repository comprises a legacy database."**

However, **Savage** teaches **"wherein said legacy repository comprises a legacy database"** as (**Savage** Col 2, Lines 8-19).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teaching of the cited references because **Savage's** teachings would have allowed **Macleod and VanDusen** to provide a generic storage model that is suitable as a data source from which enterprise data management applications may be generated in any desired model, format, or language.

4. Claims 53-55, 60, 66-67, and 71-72 are rejected under 35 U.S.C. 103(a) as being unpatentable over **Macleod et al.** (US PG Pub No. 2003/0105770) in view of **Dennis A. VanDusen** (U.S. PG Pub No. 2003/0208397), further in view of **Savage et al.** (U.S. Patent No. 6,604,110) as applied to claims 49, 51-52, 56-59 65, 68-70, and 73, above further in view of **Varadarajan Thiruvillamalai**. (**Thiruvillamalai** hereinafter) (U.S. PG Pub No. 2004/0187100).

With respect to claim 53, **Macleod VanDusen and Savage** do not explicitly teach, **"analyzing the data to generate a set of keys associated with the data."**

However, **Thiruvillamalai** discloses **"analyzing the data to generate a set of keys associated with the data"** as a request to store an element having a data type and a key, the executing storage method code stores the element in a data store according to the key and in association with data type information (**Thiruvillamalai** Paragraph 0010).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teaching of the cited references because **Thiruvillamalai's**

teachings would have allowed **Macleod VanDusen and Savage** to return an element referenced by a key, having a specified data type by acquiring a key for the set of data.

With respect to claim 54, **Macleod** teaches **"association of values with the content instance object"** as an object instance is a collection of values, or attributes that conform to the type established by the class definition (**Macleod** Paragraph 0025).

Macleod teaches the elements of claim 54 as noted above but does not explicitly teach **"generating values for the set of keys for each of the content instance objects."**

However, **Thiruvillamalai** discloses **"generating values for the set of keys for each of the content instance objects"** as the get method code determines whether the data type for the requested element matches the type index stored with the element referenced by the given key. If so, the Get method returns the data of the requested element (its value) from the data store (**Thiruvillamalai** Paragraph 0009). The Put method maintains a type index in association with each element (object) stored in the data store. The Get method validates that the type of object that was requested in the call to the Get method matches the object type that was stored in the Put method (**Thiruvillamalai** Paragraph 0007).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teaching of the cited references because **Thiruvillamalai's** teachings would have allowed **Macleod VanDusen and Savage** to return an element referenced by a key, having a specified data type by acquiring a key for the set of data.

Claims 66-67 and 71-72 are same as claims 53-54 and are rejected for the same reasons as applied hereinabove.

With respect to claim 55 **Macleod** teaches **"querying the content repository"** as allocated object elements in a database that are unused may become problematic and contribute to wasted data storage space and in some cases, decreased database query response times (**Macleod** Paragraph 0055).

Macleod teaches the elements of claim 55 as noted above but does not explicitly disclose, **"acquiring the values."**

However, **Thiruvillamalai** discloses, **"acquiring the values"** as the get method code determines whether the data type for the requested element matches the type index stored with the element referenced by the given key. If so, the Get method returns the data of the requested element (its value) from the data store (**Thiruvillamalai** Paragraph 0009). The Put method maintains a type index in association with each element (object) stored in the data store. The Get method validates that the type of object that was requested in the call to the Get method matches the object type that was stored in the Put method (**Thiruvillamalai** Paragraph 0007).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teaching of the cited references because **Thiruvillamalai's** teachings would have allowed **Macleod VanDusen and Savage** to return an element referenced by a key, having a specified data type by acquiring a key for the set of data.

With respect to claim 60, **Macleod** teaches, **"wherein the content instance objects are stored at a location remote from the content repository"** as computer 730 may operate in a networked environment using logical connections to one or more remote computers, such as a remote computer 782. Remote computer 782 may include many or all of the elements and features described herein relative to computer 730 (**Macleod** Paragraph 0100).

Macleod teaches the elements of claim 60 as noted above but does not explicitly disclose, **"wherein the content instance objects are stored at a location remote from the content repository."**

However, **Thiruvillamalai** discloses **"wherein the content instance objects are stored at a location remote from the content repository"** as in the present invention, the computer system 110 may comprise source machine from which data is being migrated, and the remote computer 180 may comprise the destination machine (**Thiruvillamalai** Paragraph 0025).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teaching of the cited references because **Thiruvillamalai's** teachings would have allowed **Macleod VanDusen and Savage** to provide storage of data in remote locations and to return an element referenced by a key, having a specified data type by acquiring a key for the set of data.

Response to Arguments

5. Applicant's arguments have been considered but are moot in view of the new ground(s) of rejection.

Examiner has combined a new secondary reference Savage et al. which discloses the argued limitation of "analyzing a set of legacy data residing in a legacy repository external to said content management system" as analyze the source databases to identify the type and form of the source data and the structure and operational system of the source databases, to determine their relevance to the EDM target database; (ii) load the source data from the legacy databases to an intermediate database using extract, transform, and load (ETL) software tools or hand-coded programs; (iii) massage, cleanse, and manipulate ("transform") the source data in the intermediate database into the form needed for the EDM application; and (iv) load the transformed source data into the EDM application, and format it in those schemas necessary for OLAP or other EDM applications (Savage Col 2, Lines 8-19 and figure 1). Therefore these lines teach analyzing legacy data and formatting this legacy data into schemas required for OLAP or other enterprise data management applications. Figure 1 shows the source legacy repository external to the enterprise data management system.

Claims must be given the broadest reasonable interpretation during examination and limitations appearing in the specification but not recited in the claim are not read into the claim (See M.P.E.P. 2111 [R-I]).

Contact Information

6. Any inquiry concerning this communication or earlier communications from the examiner should be directed to USMAAN SAEED whose telephone number is (571)272-4046. The examiner can normally be reached on M-F 8-5.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Hosain Alam can be reached on (571)272-3978. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Usmaan Saeed/
Examiner, Art Unit 2166
April 10, 2009

Usmaan Saeed
Patent Examiner
Art Unit: 2166

Hosain Alam
Supervisory Patent Examiner, Art Unit 2166

/Srirama Channavajjala/
Primary Examiner, Art Unit 2166